Open Source IC Design and Hardware Reverse Engineering or: How I learned to stop worrying and love reverse engineering RISC-V designs.

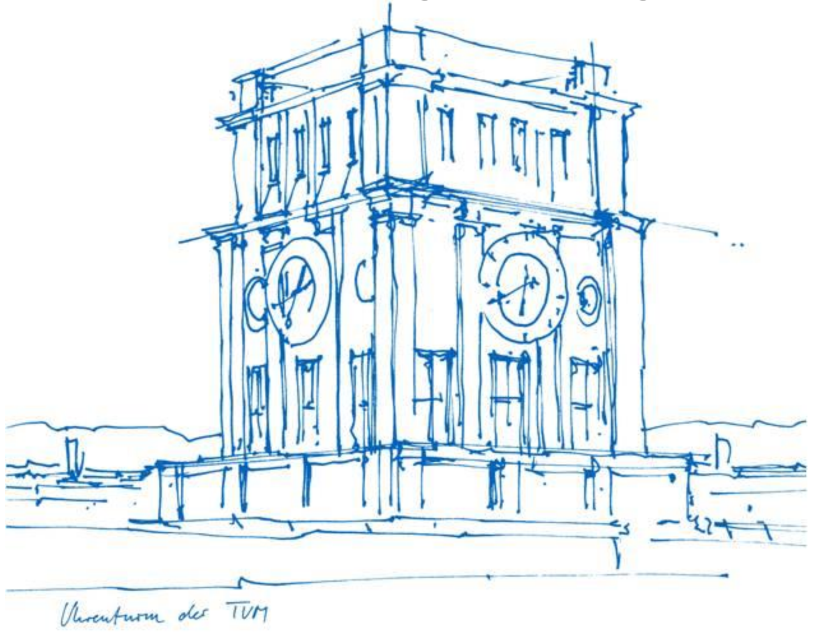
Johanna Baehr

Technische Universität München

Department of Electrical and Computer Engineering

Chair of Security in Information Technology

5th May 2022

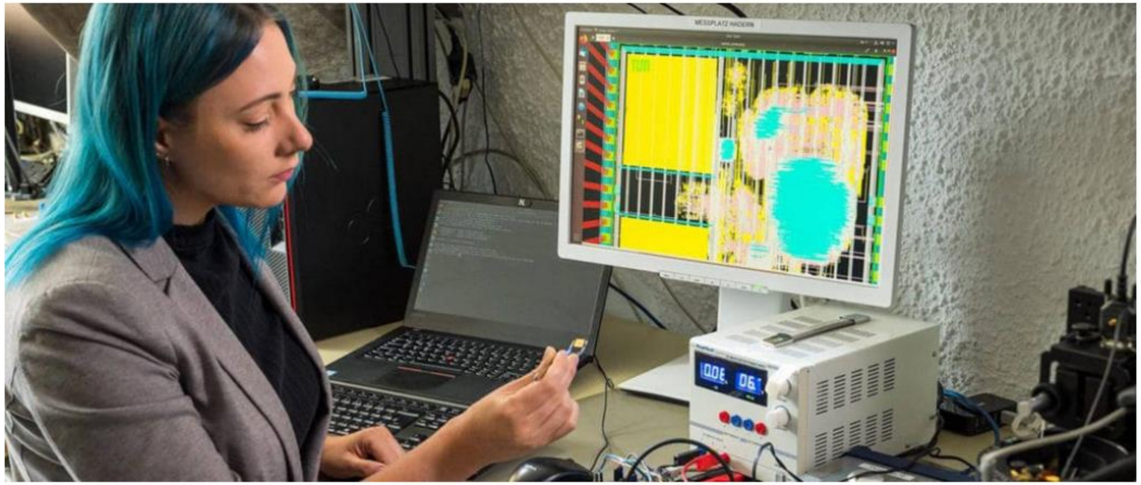


New quantum chip, incorporated with hardware trojans

Chip with secure encryption will help in the fight against hackers.

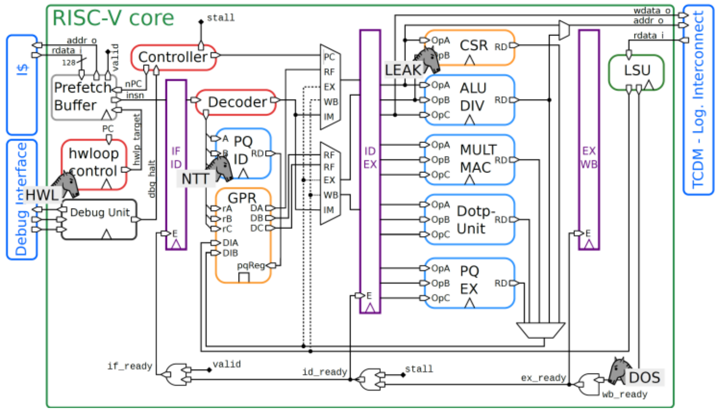
BY ASHWINI SAKHARKAR / AUGUST 5, 2021 / TECHNOLOGY

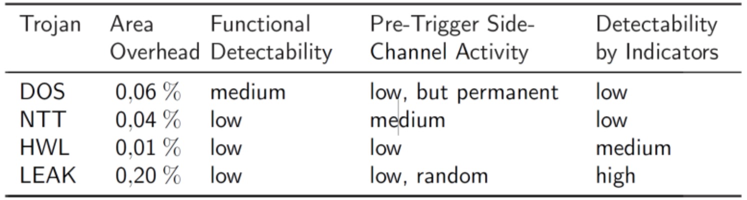
Follow us on Google News



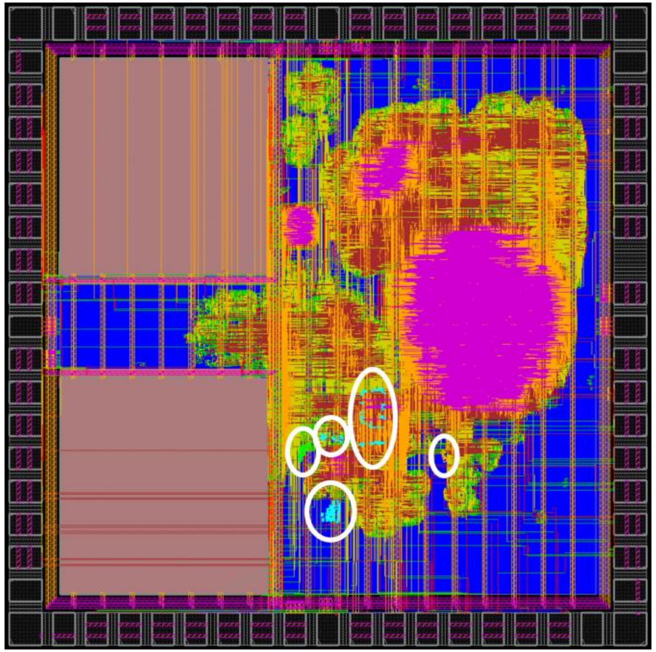
A team at the Chair of Security in Information Technology has developed a chip with particularly secure encryption technology. Johanna Baehr heads a second team at the chair that has hidden four hardware Trojans on this chip - malicious functions that are integrated directly into the circuits. Image: Astrid Eckert / TUM

A RISC-V Chip with Hardware Trojans



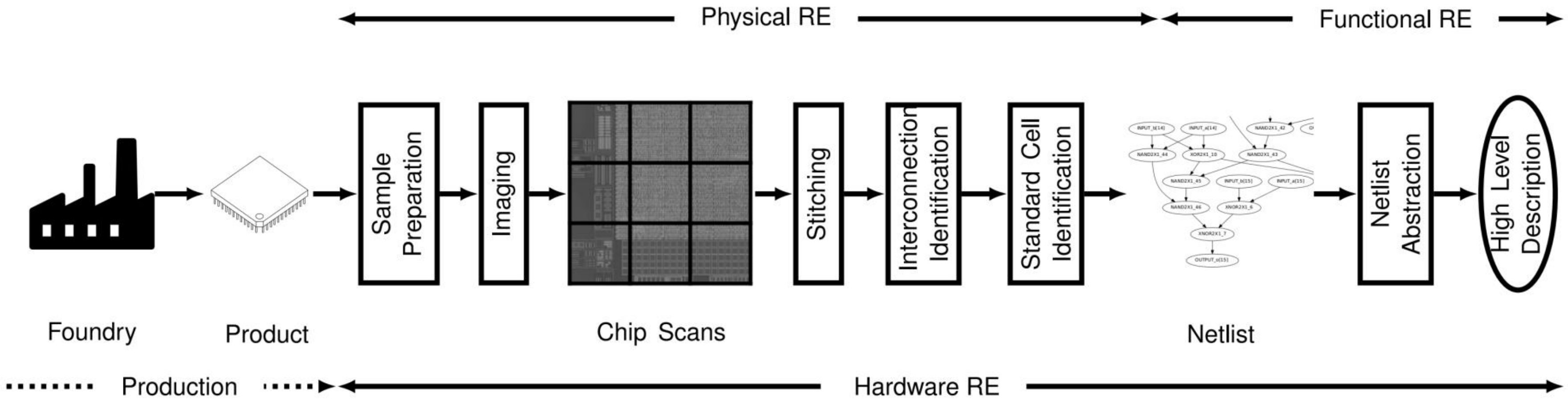


Trojan	Area Overhead	Functional Detectability	Pre-Trigger Side-Channel Activity	Detectability by Indicators
DOS	0,06 %	medium	low, but permanent	low
NTT	0,04 %	low	medium	low
HWL	0,01 %	low	low	medium
LEAK	0,20 %	low	low, random	high

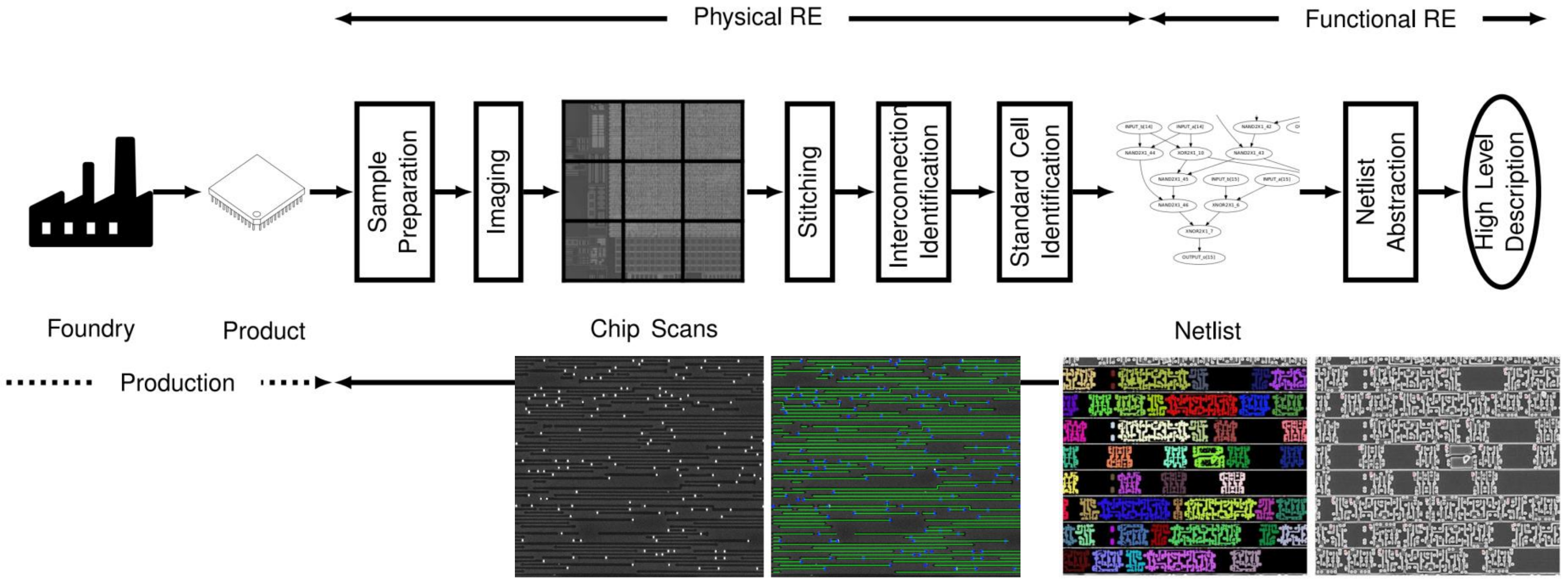


A. Hepp, et.al. Tapeout of a RISC-V crypto chip with hardware trojans: a case-study on trojan design and pre-silicon detectability. 2021

Full Hardware Reverse Engineering Process



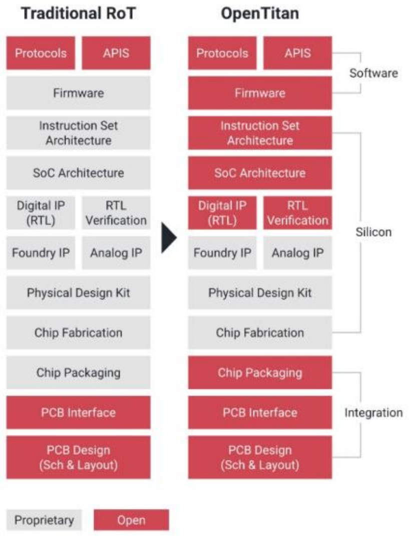
Full Hardware Reverse Engineering Process



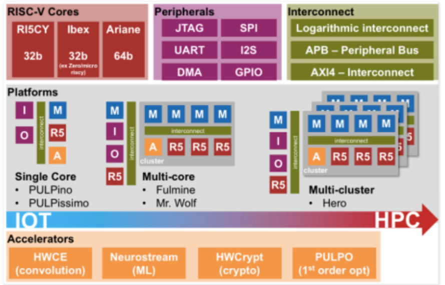
O. Thomas, On the Impact of Automating the IC Analysis Process. Blackhat 2015
B. Lippmann et.al., Verification of physical designs using an integrated reverse engineering flow for nanoscale technologies. 2019

Open-Source (ASIC) Design is here to stay

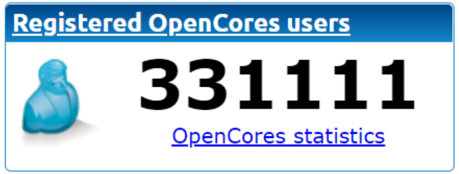
- ✓ Transparent
- ✓ Verifiable
- ✓ Customisable
- ✓ Low Cost
- ✓ Available for teaching and academia



© 2020 OpenTitan



© PULP platform, 2020


Registered OpenCores users

331111

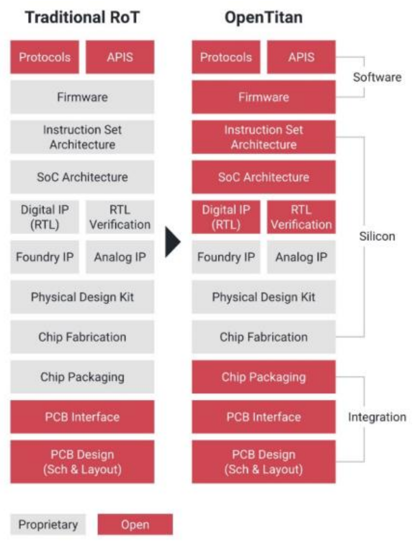
OpenCores statistics

© 1999-2021 OpenCores.org

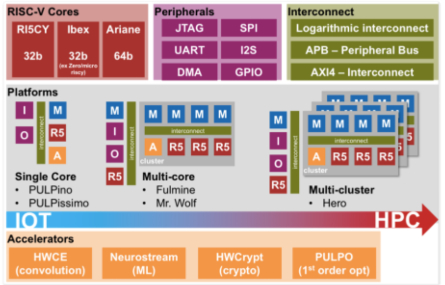
…and many more …

Open-Source (ASIC) Design is here to stay

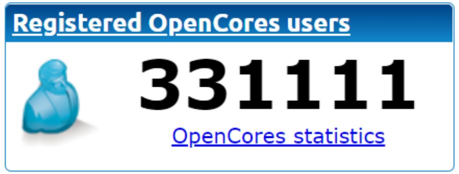
- ✓ Transparent
- ✓ **Verifiable**
- ✓ Customisable
- ✓ Low Cost
- ✓ Available for teaching and academia



© 2020 OpenTitan



© PULP platform, 2020


Registered OpenCores users

331111

OpenCores statistics

© 1999-2021 OpenCores.org

…and many more …

Hardware Reverse Engineering Goals

Identify:

- Hardware Trojans
- Product Overproduction
- IP Infringement

Hardware Reverse Engineering Goals

Identify:

- Hardware Trojans
- Product Overproduction
- IP Infringement

Hardware Reverse Engineering Goals

Identify:

- Hardware Trojans
- Product Overproduction
- IP Infringement

But also identify:

- Hardware Trojan insertion points
- IP advancement by competitors
- Possible weakness for subsequent hardware attacks

Hardware Reverse Engineering Goals

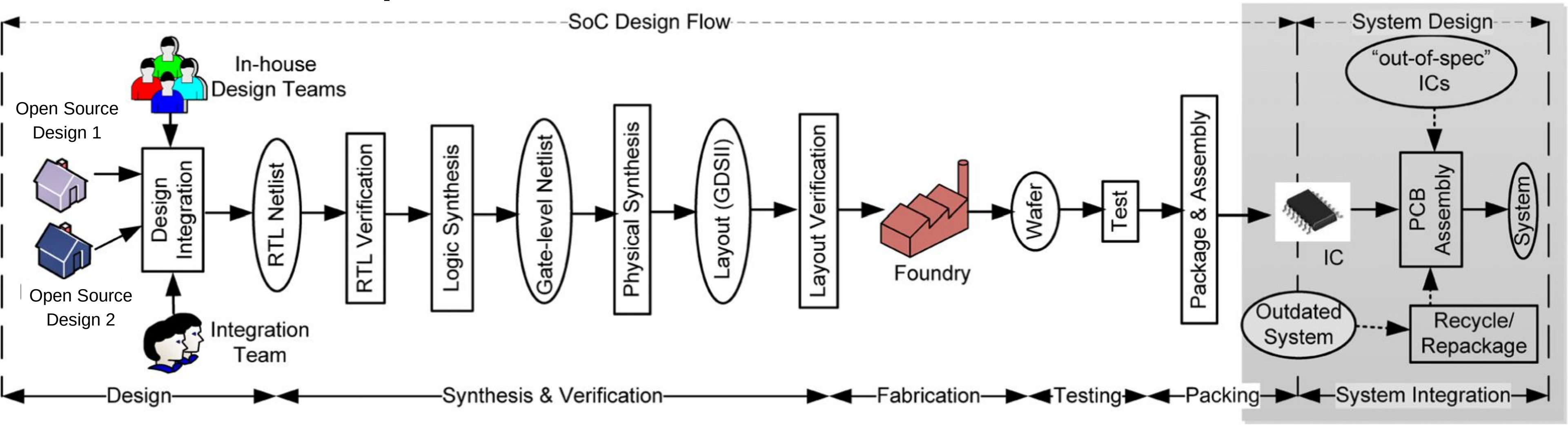
Identify:

- Hardware Trojans
- Product Overproduction
- IP Infringement

But also identify:

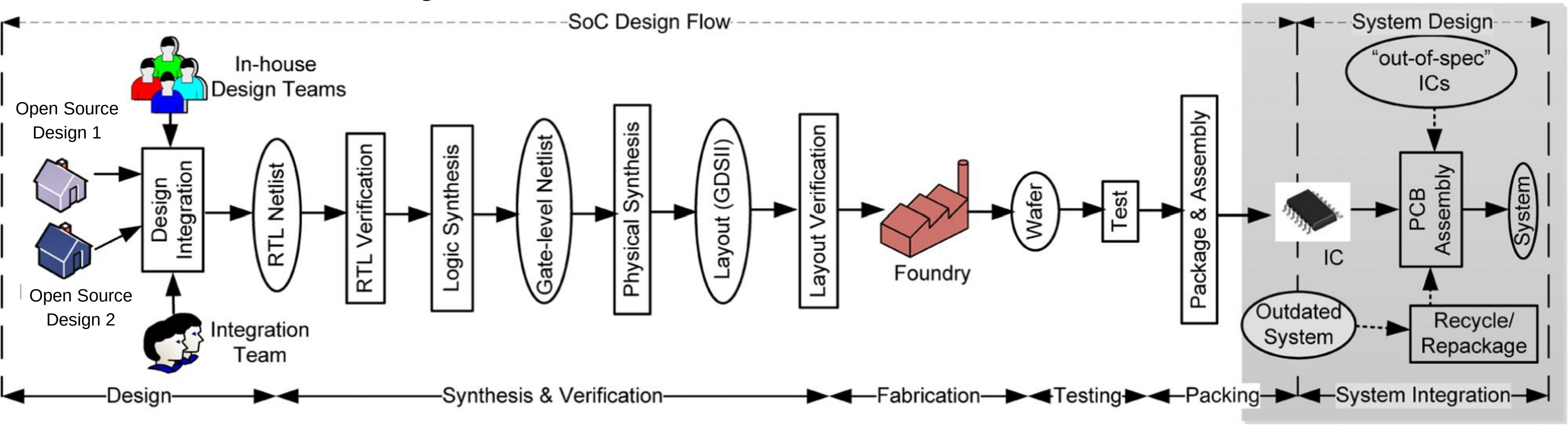
- Hardware Trojan insertion points
- IP advancement by competitors
- Possible weakness for subsequent hardware attacks

Life of a Chip



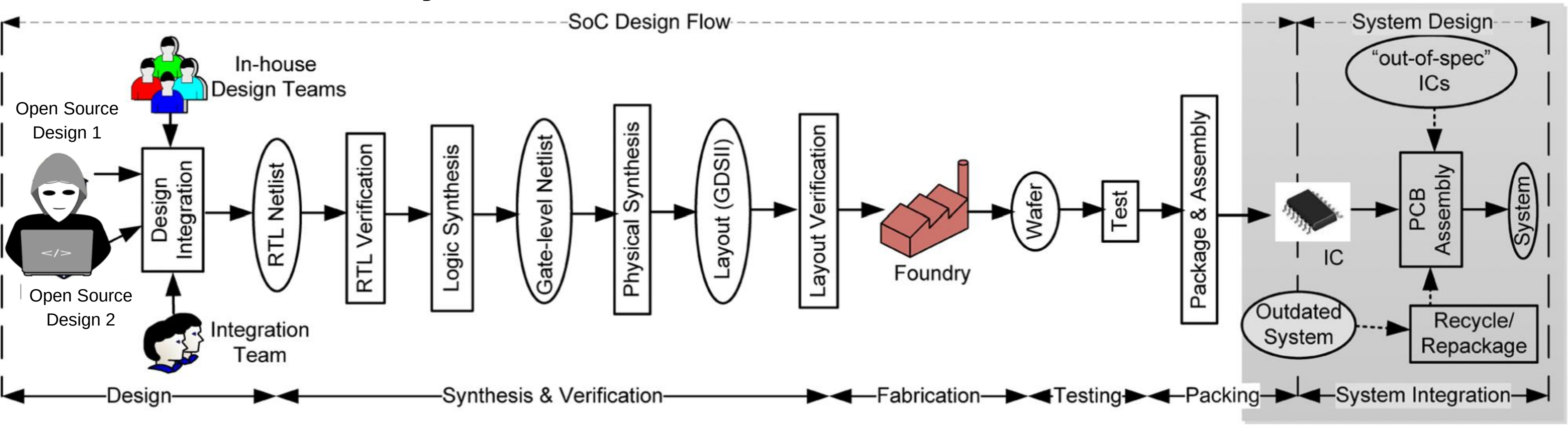
M. Rostami, et.al. Hardware security: Threat models and metrics. 2013 ICCAD. Emphasis added.

Hardware Trojan Insertion Points



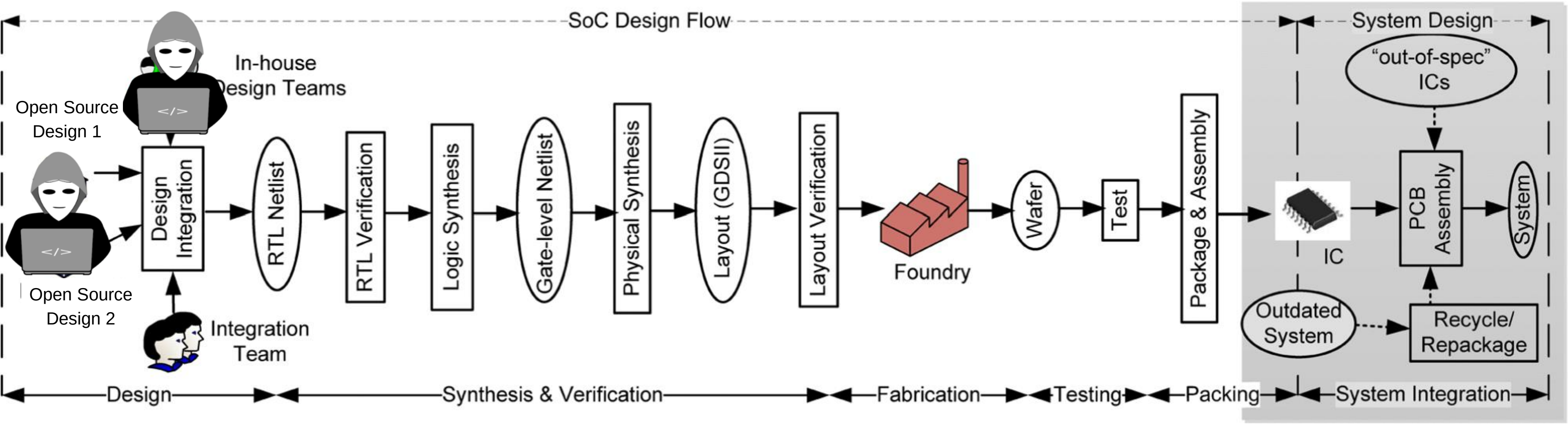
M. Rostami, et.al. Hardware security: Threat models and metrics. 2013 ICCAD. Emphasis added.

Hardware Trojan Insertion Points



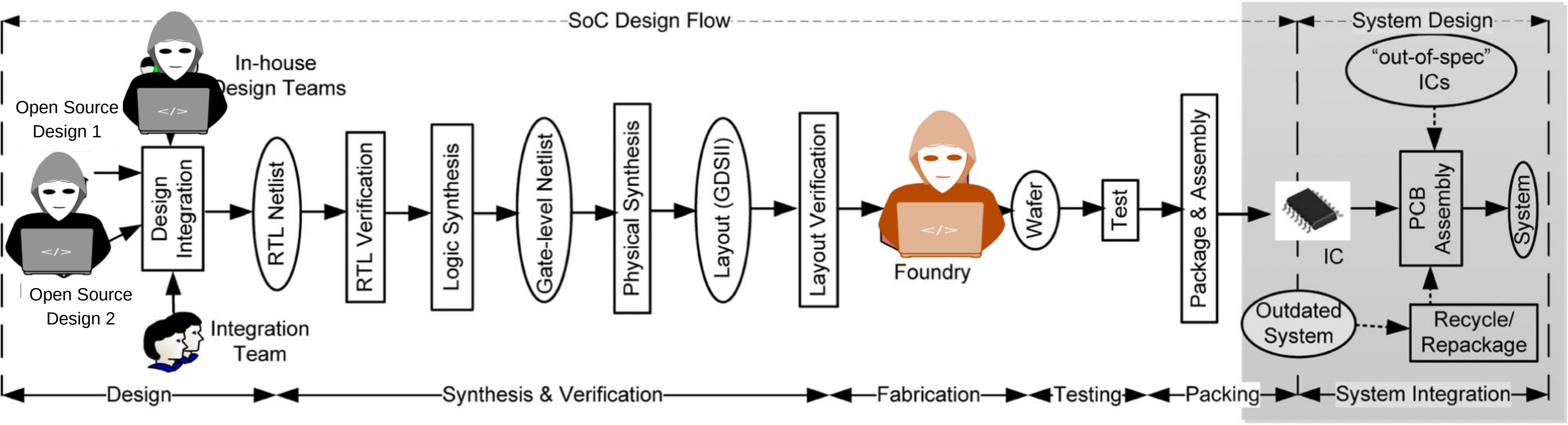
M. Rostami, et.al. Hardware security: Threat models and metrics. 2013 ICCAD. Emphasis added.

Hardware Trojan Insertion Points



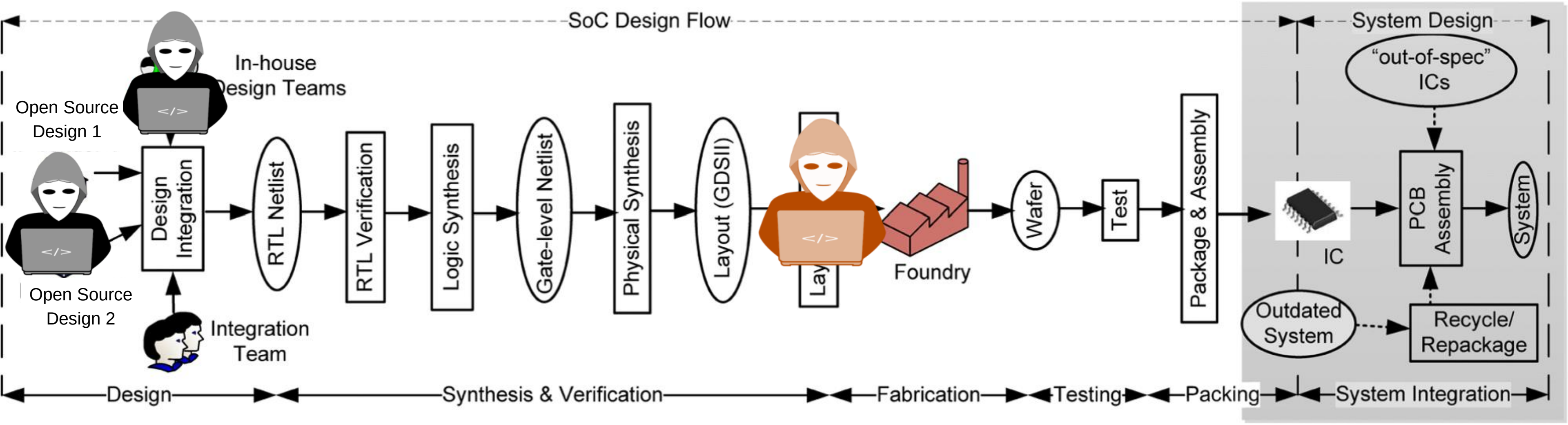
M. Rostami, et.al. Hardware security: Threat models and metrics. 2013 ICCAD. Emphasis added.

Hardware Trojan Insertion Points



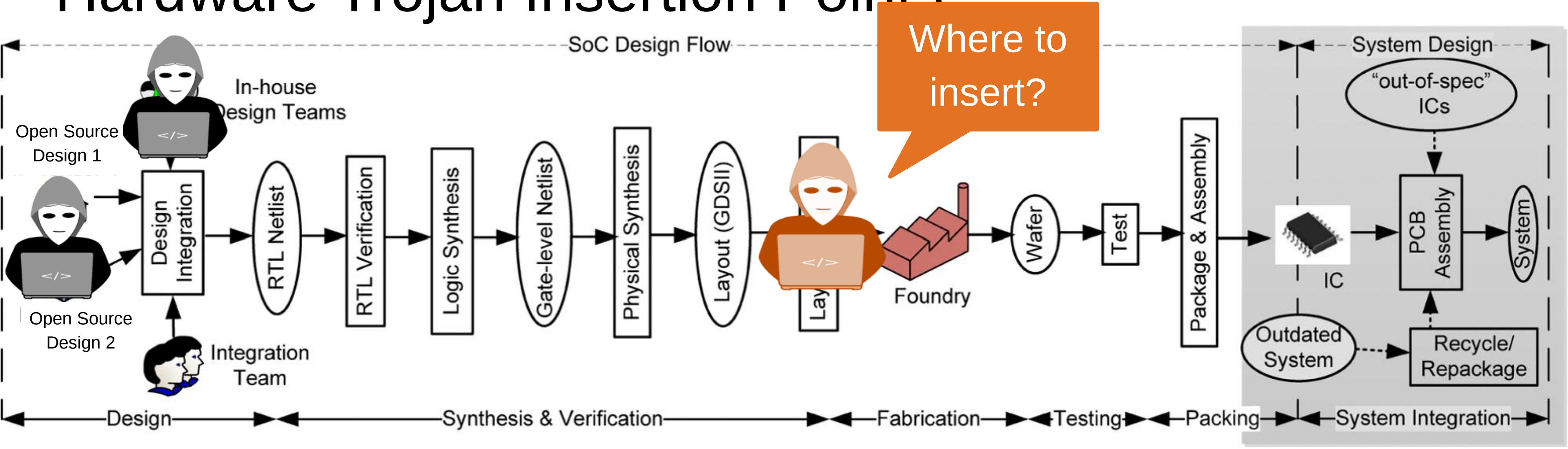
M. Rostami, et.al. Hardware security: Threat models and metrics. 2013 ICCAD. Emphasis added.

Hardware Trojan Insertion Points



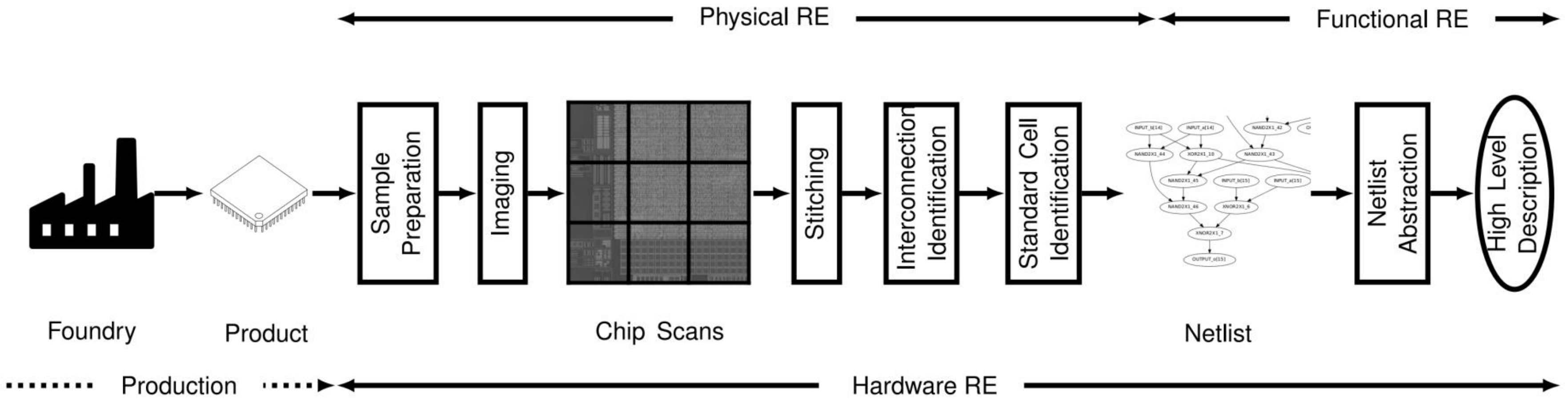
M. Rostami, et.al. Hardware security: Threat models and metrics. 2013 ICCAD. Emphasis added.

Hardware Trojan Insertion Points

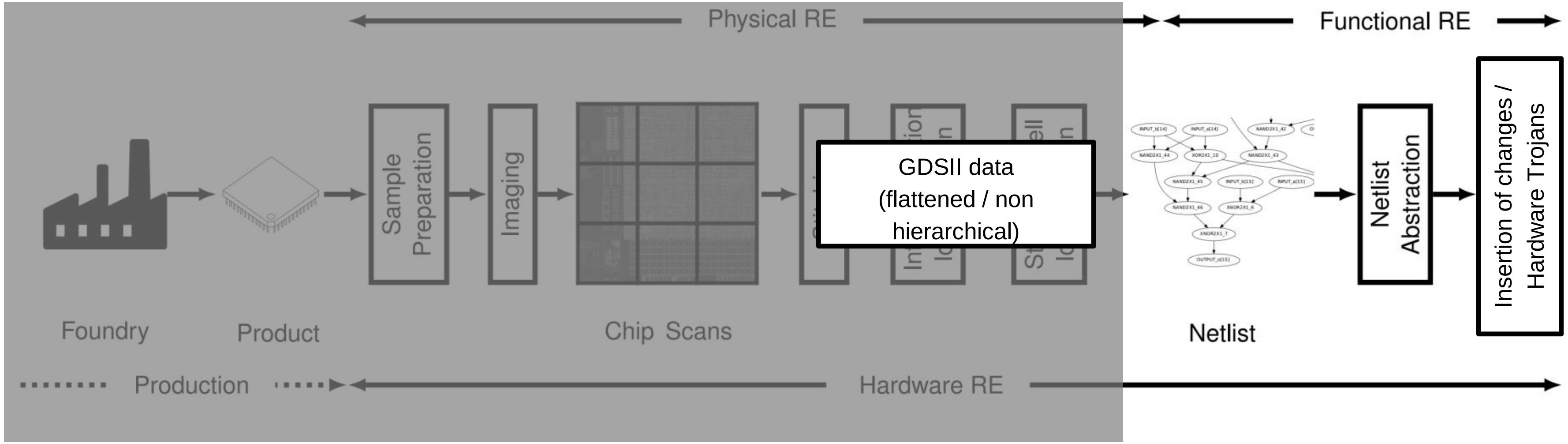


M. Rostami, et.al. Hardware security: Threat models and metrics. 2013 ICCAD. Emphasis added.

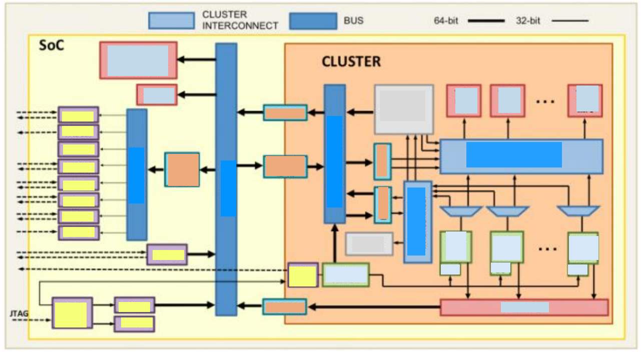
Reverse Engineering Process for Foundry



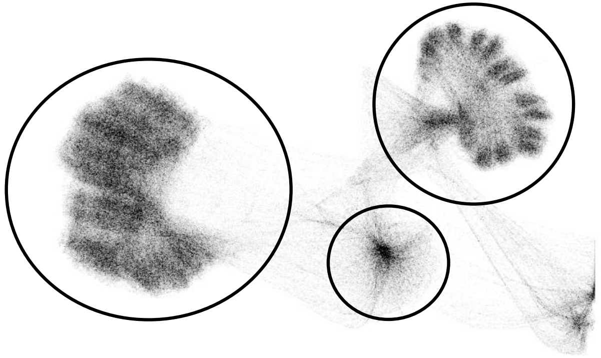
Reverse Engineering Process for Foundry



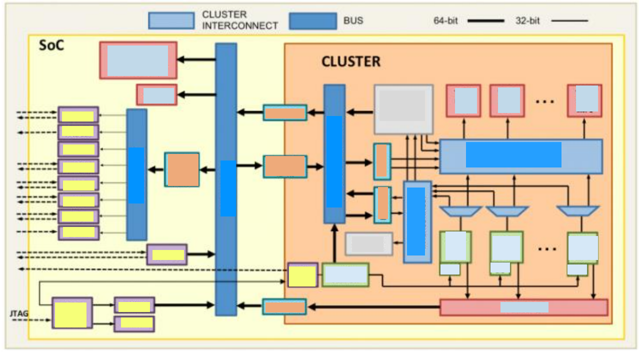
Netlist Abstraction Problems



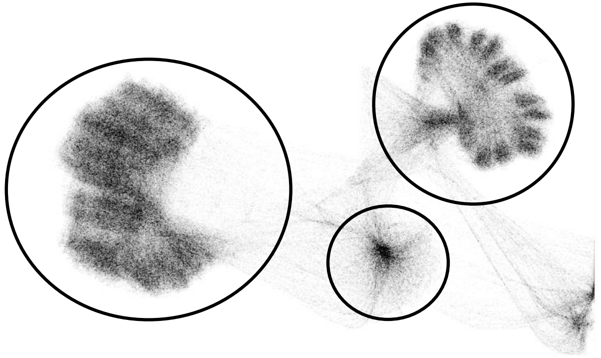
Partition

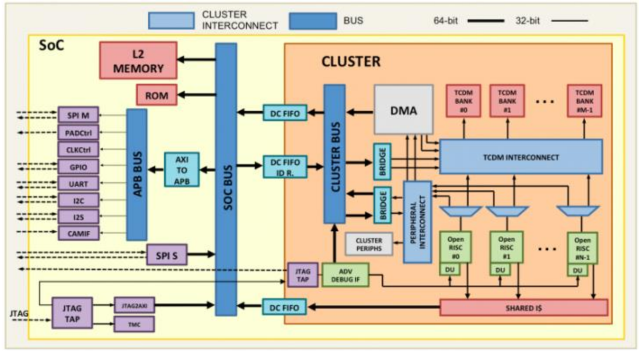


Netlist Abstraction Problems

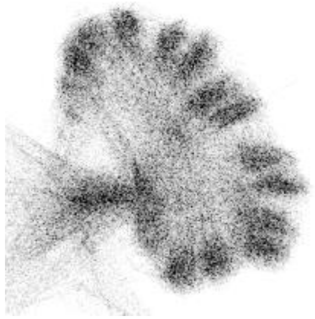


Partition





Identify

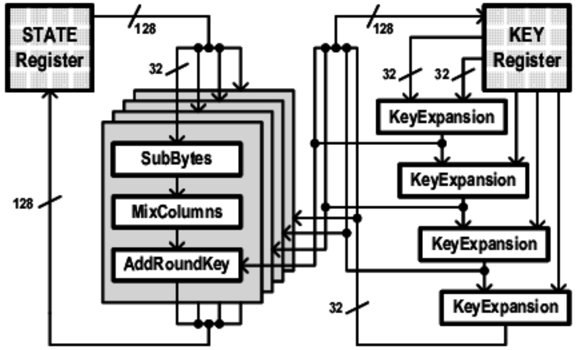


cryptographic core?
RISC-V core?

Netlist Abstraction Methods

1. Partition

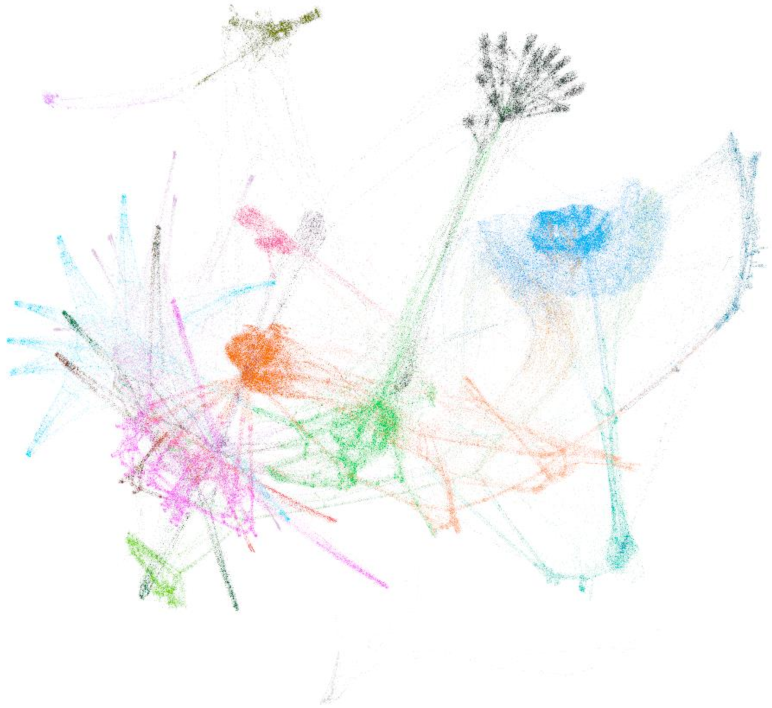
- Data Path identification (sharp)



Netlist Abstraction Methods

1. Partition

- Data Path identification (sharp)
- Structural / Graph analysis (fuzzy)



Netlist Abstraction Methods

1. Partition

- Data Path identification (sharp)
- Structural / Graph analysis (fuzzy)

2. Identify by comparison to Golden Model

Netlist Abstraction Methods

1. Partition

- Data Path identification (sharp)
- Structural / Graph analysis (fuzzy)

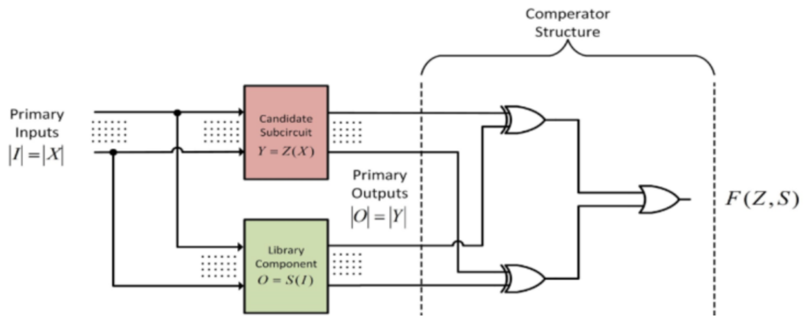
2. Identify by comparison to Golden Model

a.k.a Open Source Design

Netlist Abstraction Methods

1. Partition
- Data Path identification (sharp)
- Structural / Graph analysis (fuzzy)

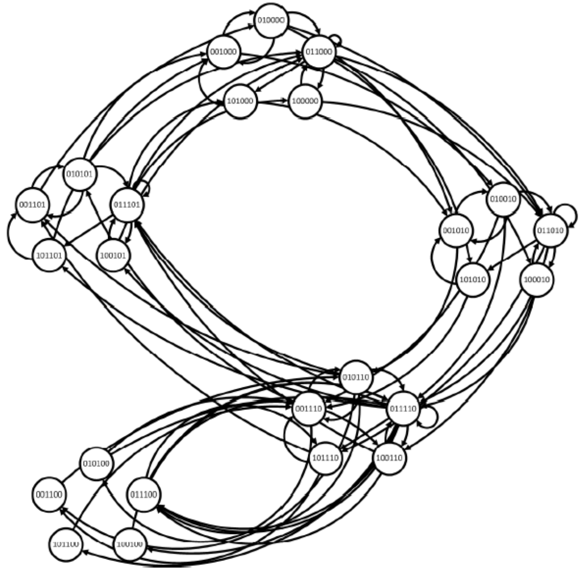
2. Identify by comparison to Golden Model
- Functional (sharp)



Netlist Abstraction Methods

1. Partition
- Data Path identification (sharp)
- Structural / Graph analysis (fuzzy)

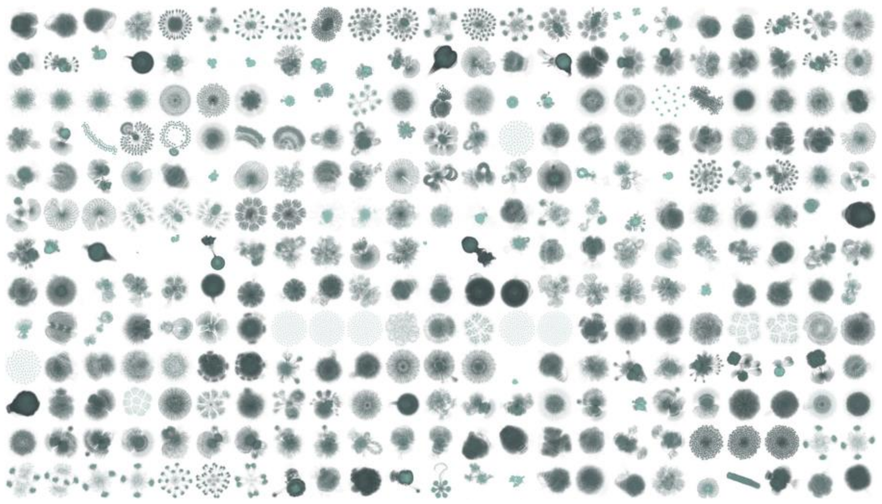
2. Identify by comparison to Golden Model
- Functional (sharp)
- Datapath-based / FSM identification



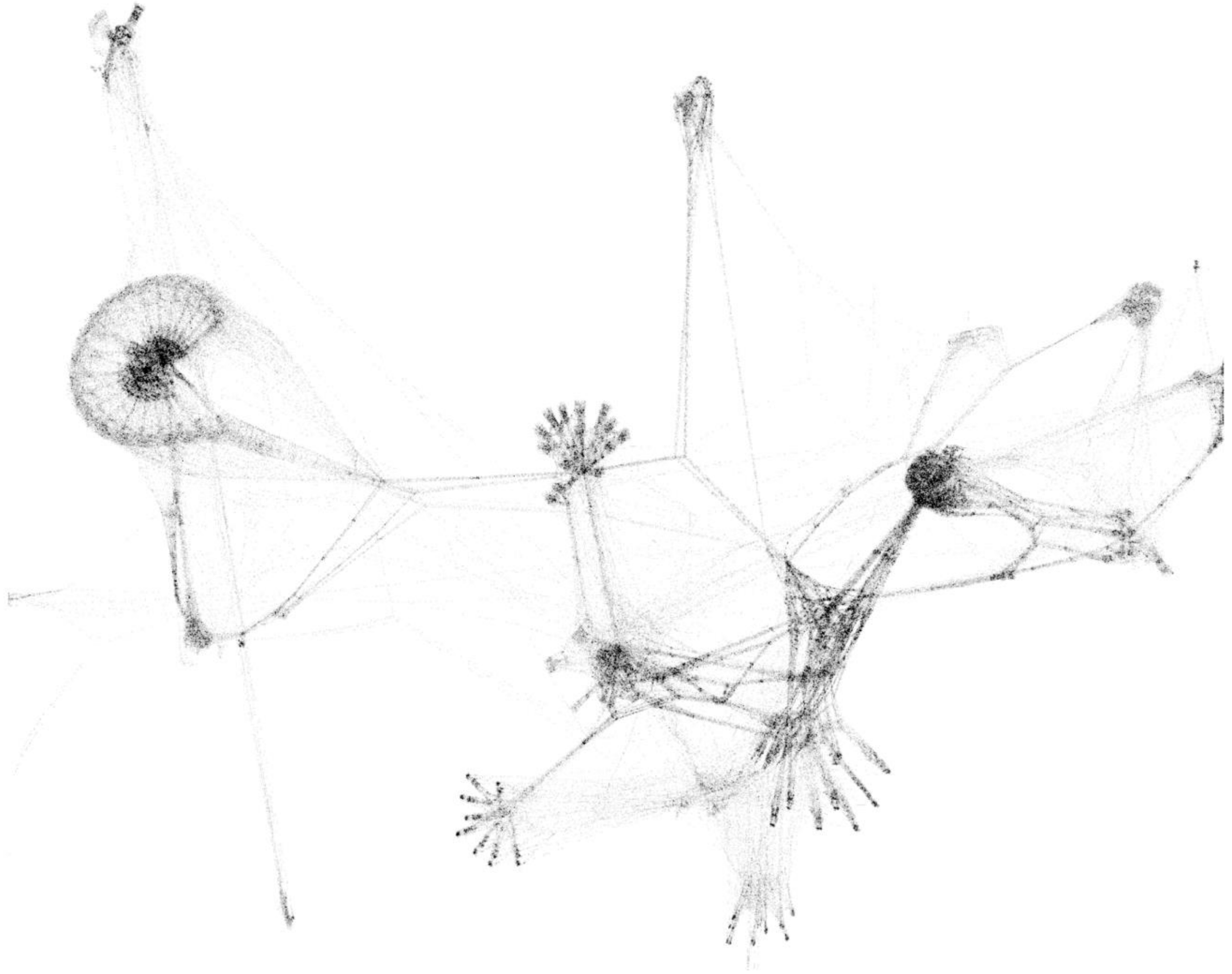
Netlist Abstraction Methods

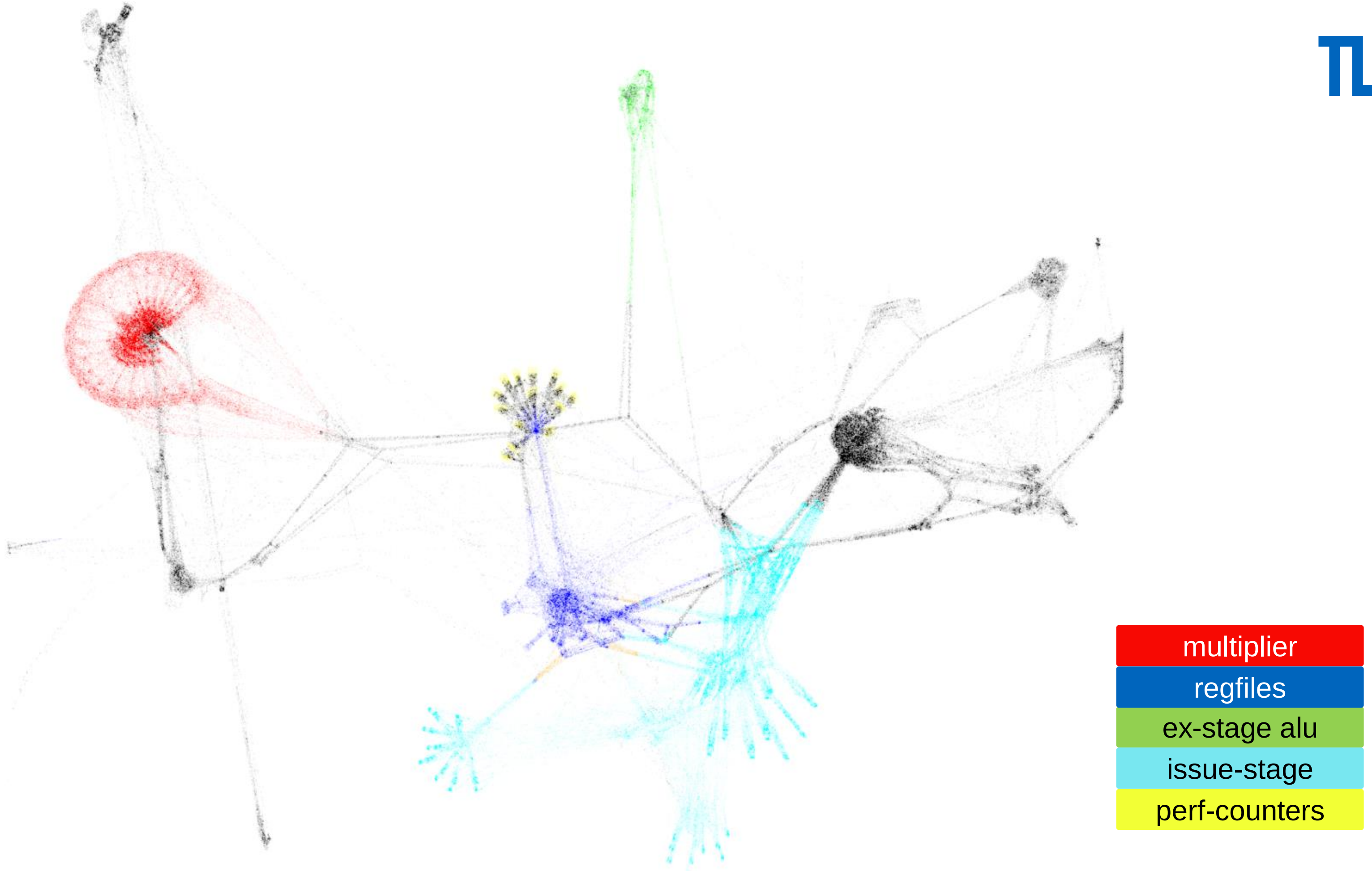
1. Partition
- Data Path identification (sharp)
- Structural / Graph analysis (fuzzy)

2. Identify by comparison to Golden Model
- Functional (sharp)
- Datapath-based / FSM identification
- Structural (fuzzy)



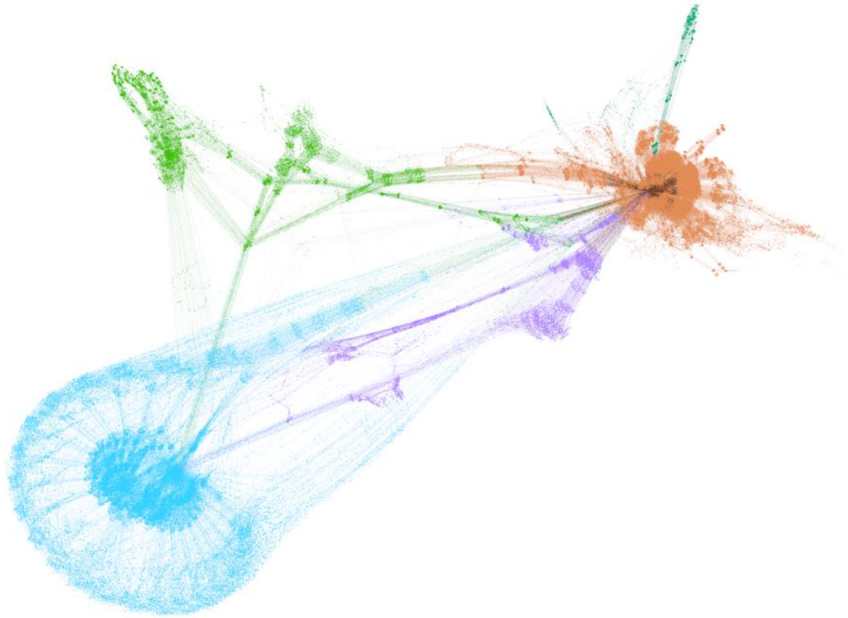
Example: RISC-V CVA6 (Ariane) Core

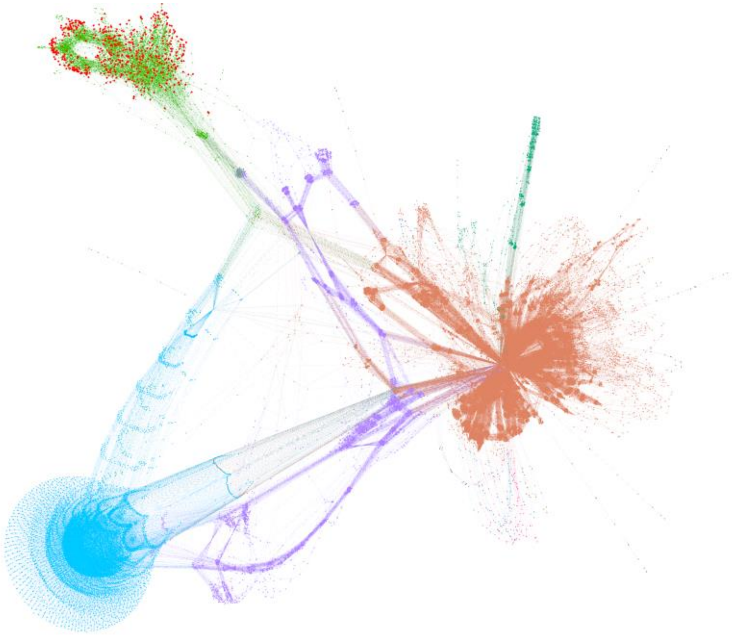



multiplier
regfiles
ex-stage alu
issue-stage
perf-counters

Example: RISC-V CVA6 (Ariane) Core

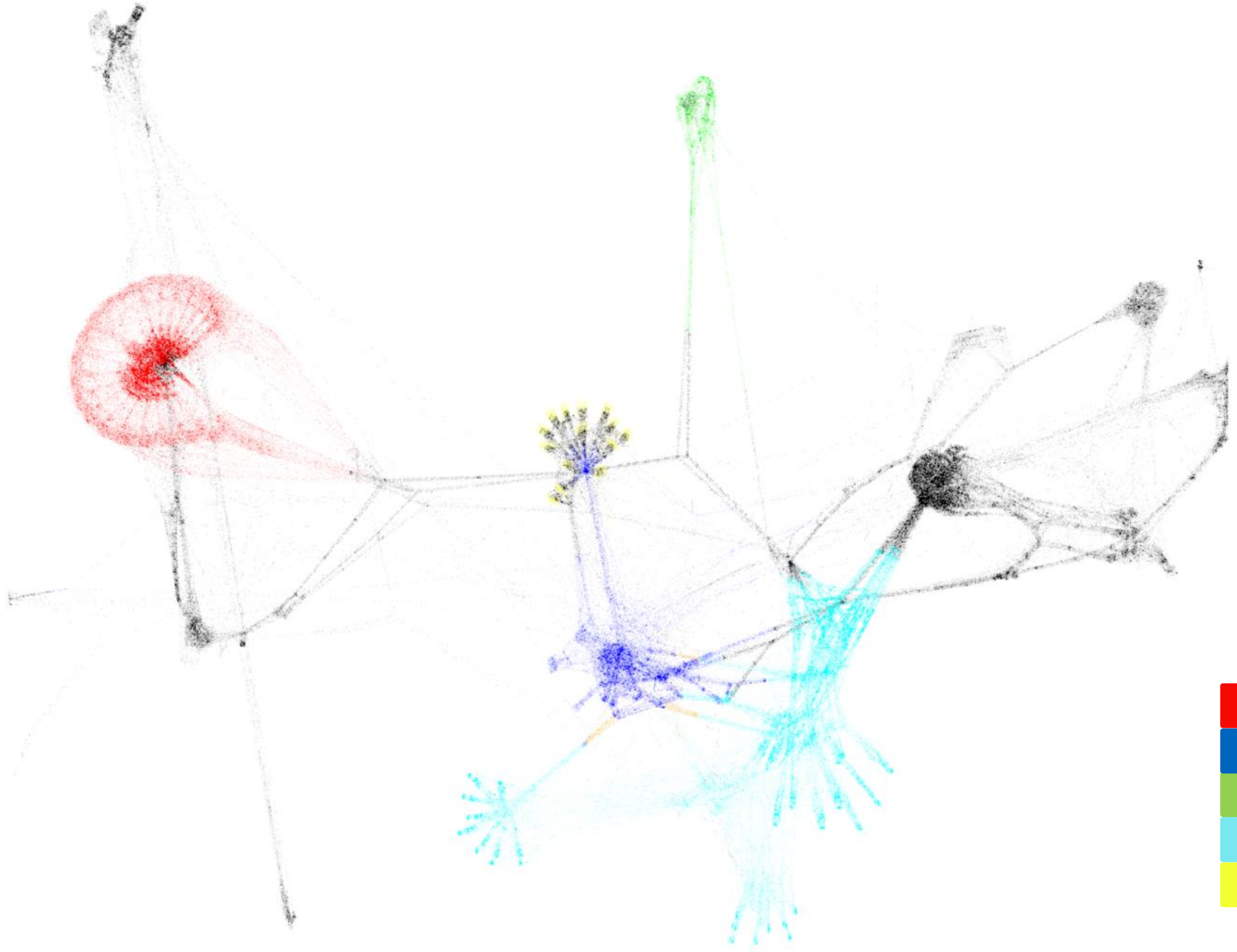
- Execution Stage with different synthesis tools, optimisations, cell libraries



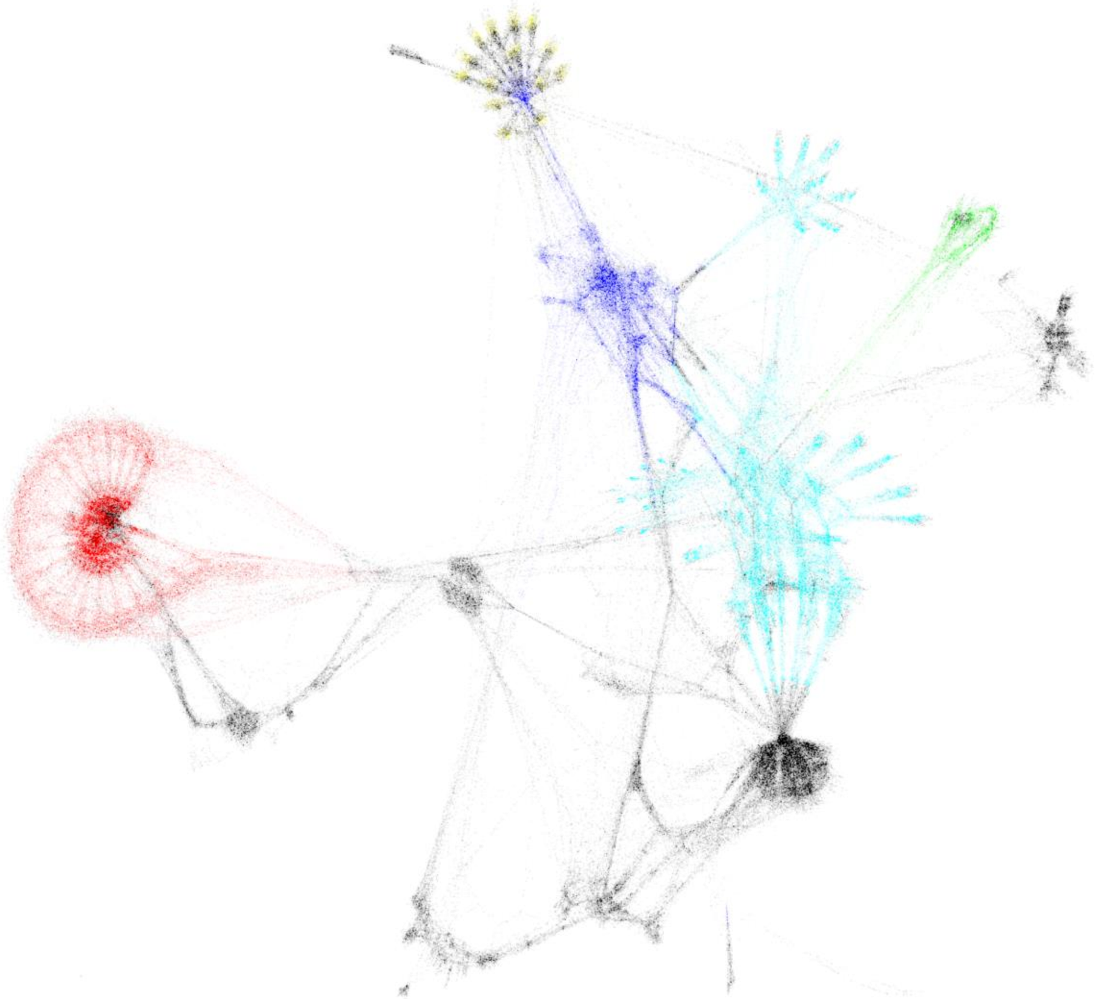


Example: RISC-V CVA6 (Ariane) Core

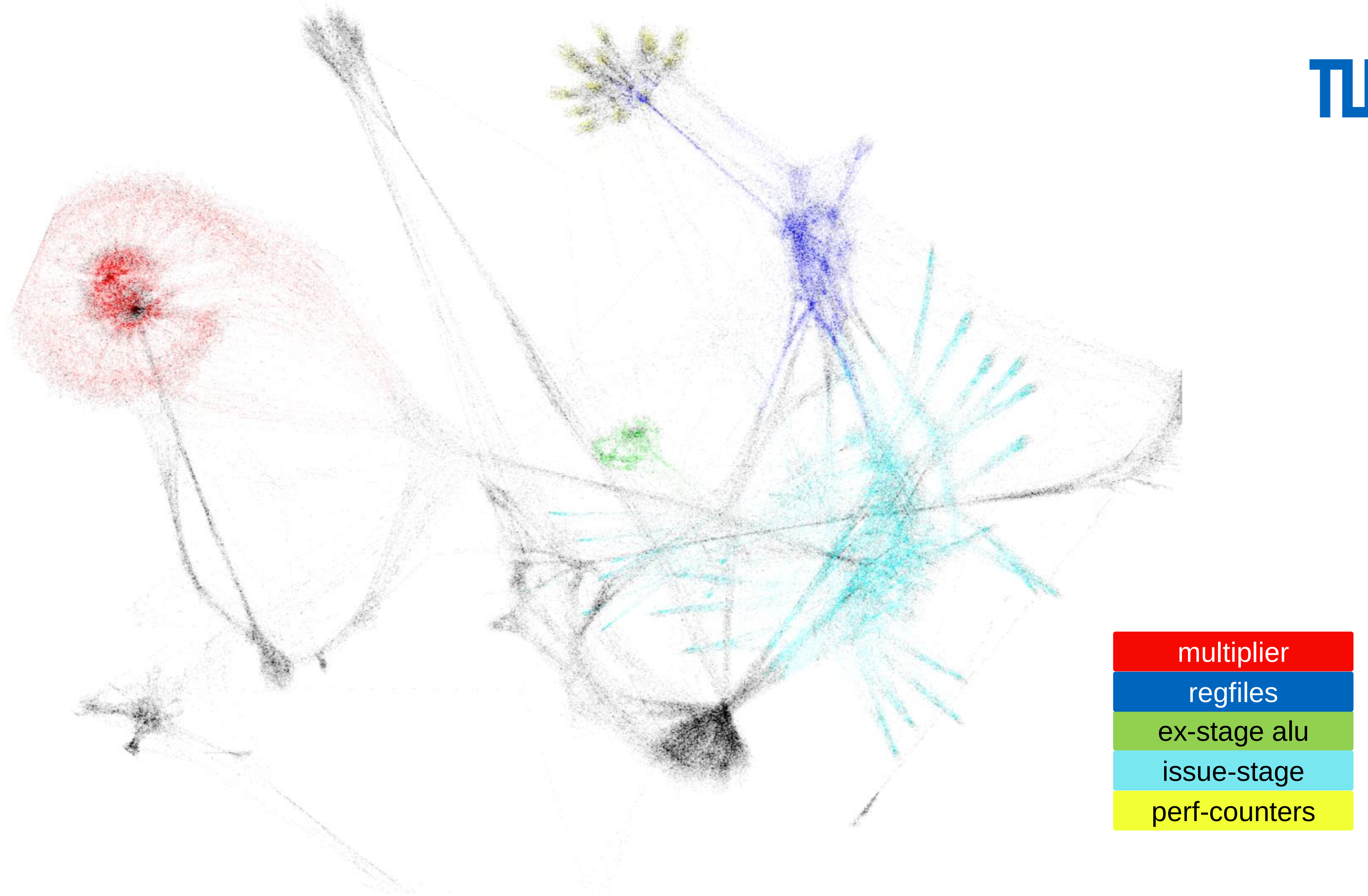
- Execution Stage with different synthesis tools, optimisations, cell libraries
- Logic Locked Core with 10%, 50% Key Gates

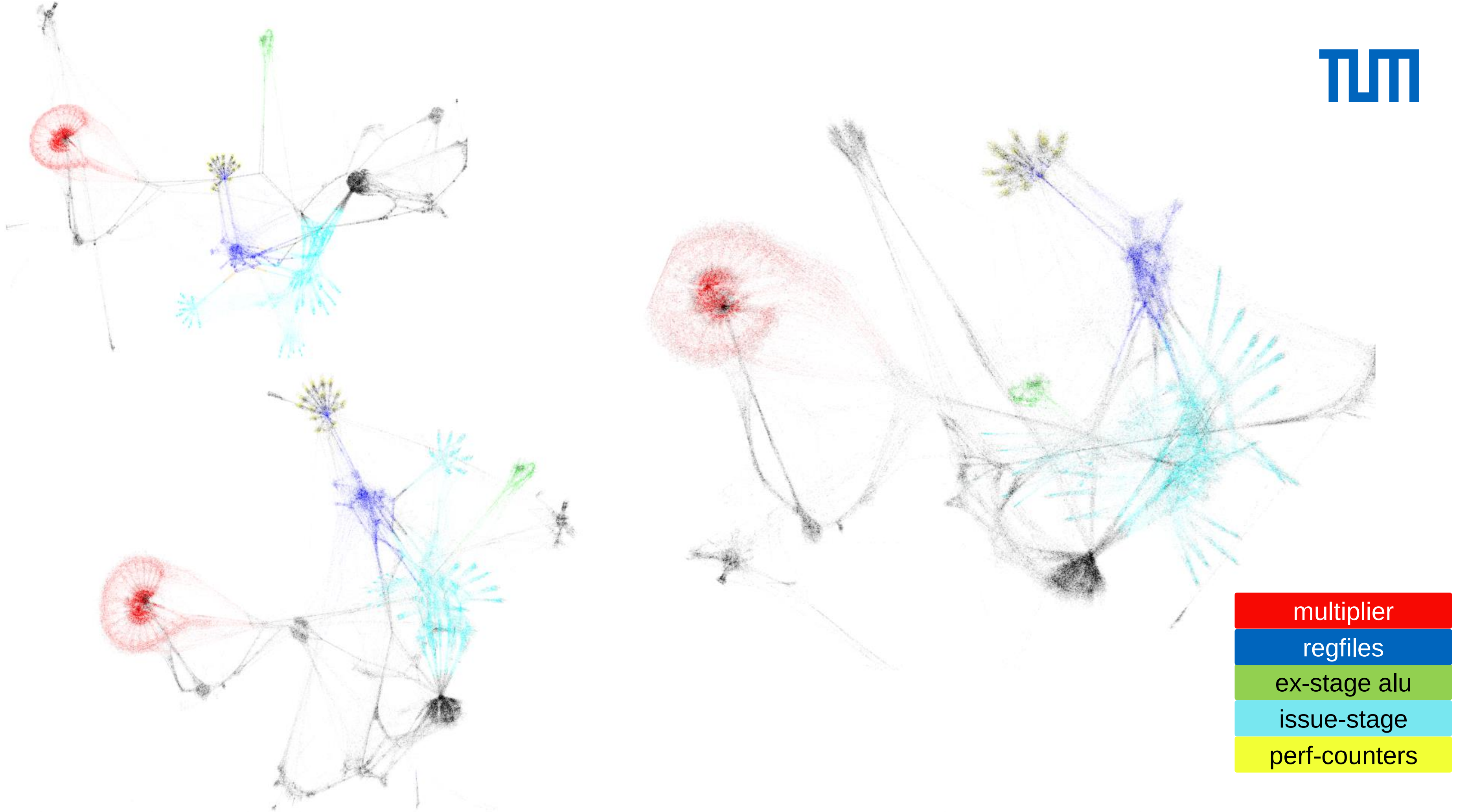


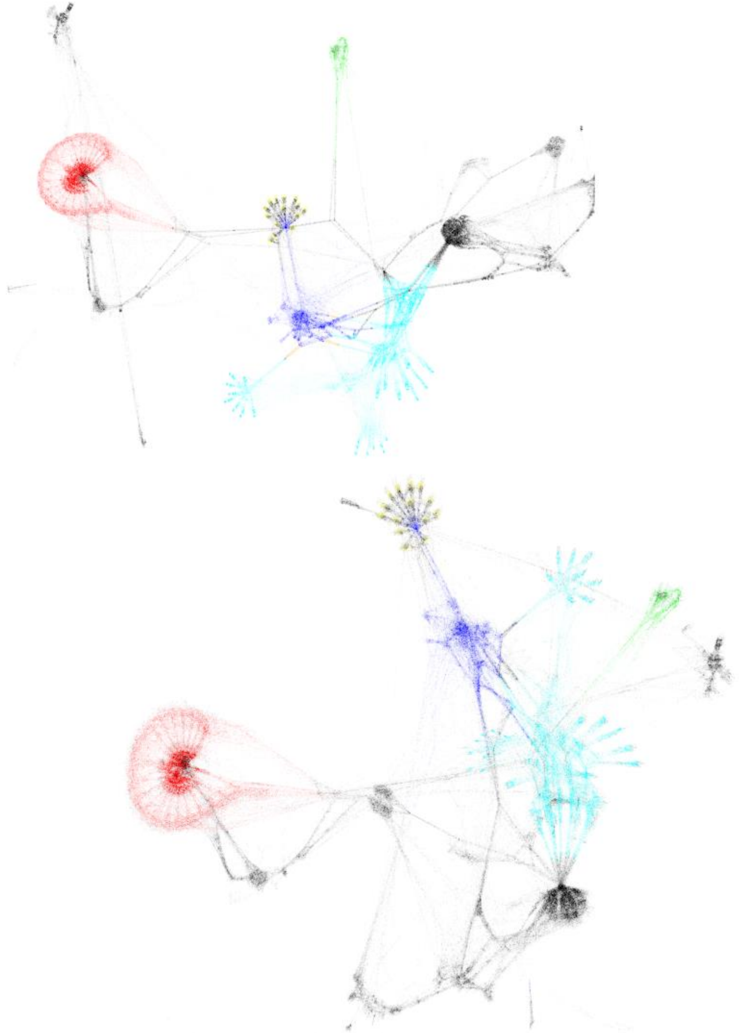
multiplier
regfiles
ex-stage alu
issue-stage
perf-counters

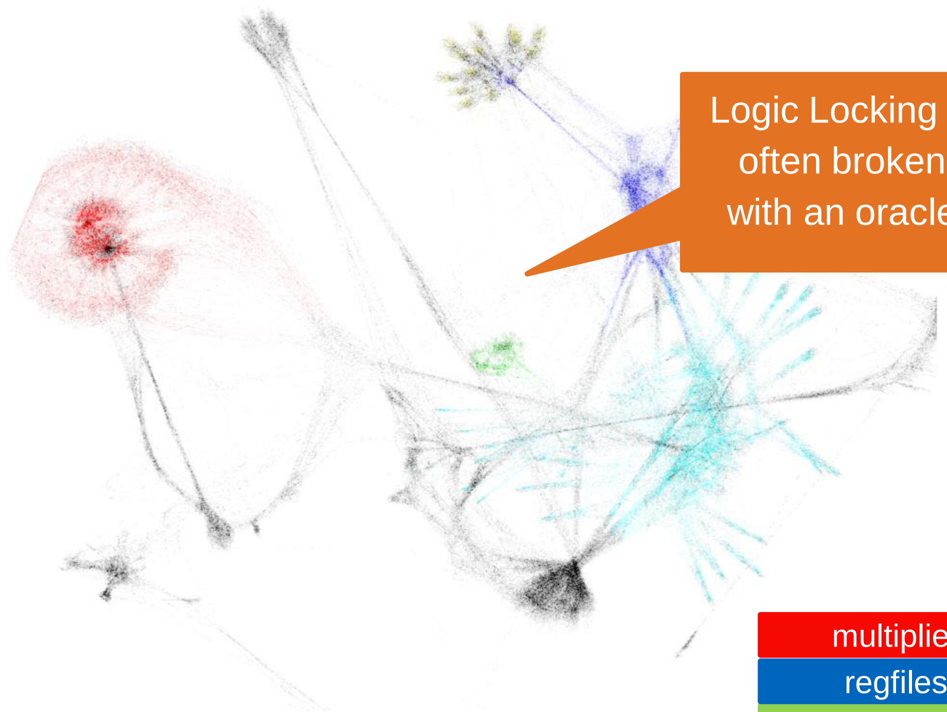


🟥 multiplier
🟦 regfiles
🟩 ex-stage alu
🟦 issue-stage
🟨 perf-counters


multiplier
regfiles
ex-stage alu
issue-stage
perf-counters


multiplier
regfiles
ex-stage alu
issue-stage
perf-counters





Logic Locking is often broken with an oracle

multiplier
regfiles
ex-stage alu
issue-stage
perf-counters

Commercial Design vs Open Source Design

	Commercial	Open Source
Partitioning is…	difficult	easy / known
Matching is….	difficult	easy / known
We can identify…	small standard designs (adders, multipliers, memory, interfaces,…) parts of commercial IP library	all submodules of design entire design
Coverage of the design is..	incomplete	incomplete, but only customised and added parts unknown
Errors occur…	often	often, but can be verified and remedied

Commercial Design vs Open Source Design

	Commercial	**Open Source**
Coverage of the design is..	incomplete	incomplete, but only customised and added parts unknown
Errors occur…	often	often, but can be verified and remedied

added parts
(cryptographic modules, own IP, etc…)
easily identifiable!

What can you do about it?

- Trust your IP / foundry (?)
- Testing / Hardware Trojan Detection (before and after manufacturing)
- Countermeasures / Logic Locking (but only if secure with oracle)
- Customization of design
- Encumber partitioning

What can you do about it?

- Trust your IP / foundry (?)
- Testing / Hardware Trojan Detection (before and after manufacturing)
- Countermeasures / Logic Locking (but only if secure with oracle)
- Customization of design
- Encumber partitioning

Open Source Design simplifies this!

Key Take Aways

- Open Source Design simplifies Reverse Engineering & Hardware Trojan Insertion (we know, because we did it)

- Solutions exist, often with little overhead

- Work on anti-reverse engineering countermeasures must continue

- Open Source Design provides a unique opportunity for **provable security against Hardware Trojan Insertion**

Thank you and Questions?